UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/037,666	01/03/2002	Sriram Vajapeyam	42390P11927	8277

8791 7590 09/22/2004

BLAKELY SOKOLOFF TAYLOR & ZAFMAN
12400 WILSHIRE BOULEVARD
SEVENTH FLOOR
LOS ANGELES, CA 90025-1030

EXAMINER	
HUISMAN, DAVID J	

ART UNIT	PAPER NUMBER
2183	

DATE MAILED: 09/22/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)
	10/037,666	VAJAPEYAM ET AL.
	Examiner	Art Unit
	David J. Huisman	2183

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *03 January 2002*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-30* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-30* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *03 January 2002* is/are: a)☐ accepted or b)☒ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *08 April 2002*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-30 have been examined.

Papers Submitted

2. It is hereby acknowledged that the following papers have been received and placed of record in the file: IDS as received on 4/8/2002.

Specification

3. The title of the invention is not descriptive. A new title is required that is clearly indicative of the invention to which the claims are directed.

4. The disclosure is objected to because of the following informalities: On page 7, paragraph 25, replace "producer consumer" with --producer/consumer--. On page 9, paragraph 30, replace "innstruction" with --instruction--. On page 17, paragraph 55, insert a comma after 201 and 202.

Appropriate correction is required.

Drawings

5. The drawings are objected to as failing to comply with 37 CFR 1.84(p)(5) because they include the following reference character(s) not mentioned in the description: In Fig.1, reference numbers 11, 12, and 13 must be mentioned. Corrected drawing sheets, or amendment to the specification to add the reference character(s) in the description, are required in reply to the Office action to avoid abandonment of the application. Any amended replacement drawing sheet

should include all of the figures appearing on the immediate prior version of the sheet, even if only one figure is being amended. The replacement sheet(s) should be labeled "Replacement Sheet" in the page header (as per 37 CFR 1.84(c)) so as not to obstruct any portion of the drawing figures. If the changes are not accepted by the examiner, the applicant will be notified and informed of any required corrective action in the next Office action. The objection to the drawings will not be held in abeyance.

6. The drawings are objected to because of the following minor informalities: In Fig.4, it is requested that the applicant move labels DC1, DC2, and DC3 so that they don't touch the circles in the dependency chain (for increased clarity). Also, the specification, when referring to instructions, uses uppercase I's (as opposed to lowercase i's which are used in both Fig.3 and Fig.4). It is requested that applicant either modify the specification to use lowercase i's (for instance, i00), or Fig.3 and Fig.4 be modified to use uppercase I's (such as I00). Finally Fig.6, is not even mentioned within the detailed description. Corrected drawing sheets are required in reply to the Office action to avoid abandonment of the application. Any amended replacement drawing sheet should include all of the figures appearing on the immediate prior version of the sheet, even if only one figure is being amended. The figure or figure number of an amended drawing should not be labeled as "amended." If a drawing figure is to be canceled, the appropriate figure must be removed from the replacement sheet, and where necessary, the remaining figures must be renumbered and appropriate changes made to the brief description of the several views of the drawings for consistency. Additional replacement sheets may be necessary to show the renumbering of the remaining figures. The replacement sheet(s) should be labeled "Replacement Sheet" in the page header (as per 37 CFR 1.84(c)) so as not to obstruct any

portion of the drawing figures. If the changes are not accepted by the examiner, the applicant will be notified and informed of any required corrective action in the next Office action. The objection to the drawings will not be held in abeyance.

Claim Objections

7. Claim 10 is objected to because of the following informalities: In line 5, replace "creates" with --create--.. Appropriate correction is required.

8. Claim 20 is objected to because of the following informalities: In line 5, insert --the-- between "by" and "dependency". Appropriate correction is required.

9. Claim 28 is objected to because of the following informalities: In line 7, insert --the-- between "by" and "dependency". Appropriate correction is required.

Claim Rejections - 35 USC § 112

10. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

11. Claim 20 recites the limitation "the dependencies" in the last two lines of the claim. There is insufficient antecedent basis for this limitation in the claim.

12. Claim 22 recites the limitations "the control flow logic" and "the data flow logic". There is insufficient antecedent basis for this limitation in the claim.

13. Claim 27 recites the limitation "the selecting". There is insufficient antecedent basis for this limitation in the claim.

14. Claim 28 recites the limitation "the dependencies" in the last two lines of the claim. There is insufficient antecedent basis for this limitation in the claim.

15. Claim 30 recites the limitations "the control flow logic" and "the data flow logic". There is insufficient antecedent basis for this limitation in the claim.

Claim Rejections - 35 USC § 102

16. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

17. Claims 1-3, 8-12, 17-22, and 28-30 are rejected under 35 U.S.C. 102(e) as being anticipated by Batten et al., U.S. Patent No. 6,260,189 (herein referred to as Batten).

18. Referring to claim 1, Batten has taught a logic circuit comprising:

a) a control flow logic to select a trace descriptor for processing including at least one dependency descriptor including dependency information for each instruction sequence. See Fig.9-11 and column 6, lines 50-58, and note the "ccdd" trace descriptor. This descriptor includes a dependency descriptor, which in turn provides dependency information (!RAW, !WAW, !WAR, !pred) regarding the associated instruction trace (sequence of instructions which may be considered a unit).

b) a data flow logic coupled to the control flow logic to execute a plurality of instruction sequences according to the dependency information stored in the dependency descriptor. See

column 6, lines 44-58, and column 3, lines 52-57. Note that the dependency information is used by the hardware in executing the instructions such that stalls may be avoided by disabling the components which check for stalls.

19. Referring to claim 2, Batten has taught a logic circuit as described in claim 1. Batten has further taught a first storage area coupled to the control flow logic and the data flow logic, the first storage area to store the dependency descriptor. Note from 5, lines 47-49, that the dependency descriptors are located within a ccdd instruction, which has the format shown in Fig.7. A ccdd instruction is inherently stored in a first storage location (designated by a first memory address) where it is fetched by the system (column 1, lines 22-23), and therefore, the dependency descriptor is stored in a first storage area.

20. Referring to claim 3, Batten has taught a logic circuit as described in claim 2. Batten has further taught a second storage area coupled to the control flow logic, the second storage area to store a trace descriptor. Note from 5, lines 47-49, that the trace descriptor is a ccdd instruction, which has the format shown in Fig.7. A ccdd instruction is inherently stored in a storage location (designated by a memory address) where it is fetched by the system (column 1, lines 22-23). Clearly, if multiple ccdd instructions exist, they will be stored in different storage locations, and therefore, the trace descriptor is stored in a second storage area (that is, a trace descriptor which is not associated with the dependency descriptors from claim 2 will be stored in a second storage area).

21. Referring to claim 8, Batten has taught a logic circuit as described in claim 1. Batten has further taught that the trace descriptor includes aggregate live-in data for the at least one dependency descriptor. See Fig.9 and Fig.8, for instance. If the dependency descriptor specifies

that a WAR hazard does not exist, then the descriptor includes aggregate live-in data. That is, the descriptor comprises data which specifies that a hazard will not occur involving live-in data. And, in Fig.10, this is the case because s5 is a live-in value (from Fig.9, assuming the trace in Fig.10 follows that of Fig.9) and there is no hazard involving s5. Note that applicant has not defined within the claim what is meant by aggregate live-in data and consequently, because the examiner must take the broadest interpretation of the claim language, any data which involves live-in values is aggregate live-in data.

22. Referring to claim 9, Batten has taught a logic circuit as described in claim 1. Batten has further taught that the trace descriptor includes aggregate live-out data for the at least one dependency descriptor. See Fig.9 and Fig.8, for instance. If the dependency descriptor specifies that a RAW hazard exists, then the descriptor includes aggregate live-out data. That is, the descriptor comprises data which specifies that a hazard will occur involving live-out data. And, in Fig.9, this is the case because s5 is a live-out value (assuming the trace in Fig.10 follows that of Fig.9) and there is a hazard involving s5. Note that applicant has not defined within the claim what is meant by aggregate live-out data and consequently, because the examiner must take the broadest interpretation of the claim language, any data which involves live-out values is aggregate live-out data.

23. Referring to claim 10, Batten has taught a computer system comprising:

a) at least one memory device. See column 1, lines 22-23.

b) a bus coupled to the at least one memory device. It is inherent a bus is connected to the memory device so data is able to travel between memory and the rest of the system.

c) a control flow logic to analyze dependencies among instruction sequences and creates a dependency descriptor including dependency information for each instruction sequence. See Fig.9-11 and note that each ccdd instruction includes a dependency descriptor which specifies the types of dependencies that exist (or don't exist - see Fig.8) within the associated instruction sequence. Clearly, the instruction sequence must've been analyzed so that such a descriptor may be formed.

d) a data flow logic coupled to the control flow logic to execute a plurality of the instruction sequences according to the dependency information stored in the dependency descriptor. See column 6, lines 44-58, and column 3, lines 52-57. Note that the dependency information is used by the hardware in executing the instructions such that stalls may be avoided by disabling the components which check for stalls.

24. Referring to claim 11, Batten has taught a computer system as described in claim 10. Furthermore, claim 11 is rejected for the same reasons set forth in the rejection of claim 2.

25. Referring to claim 12, Batten has taught a computer system as described in claim 11. Furthermore, claim 12 is rejected for the same reasons set forth in the rejection of claim 3.

26. Referring to claim 17, Batten has taught a computer system as described in claim 10. Furthermore, claim 17 is rejected for the same reasons set forth in the rejection of claim 8.

27. Referring to claim 18, Batten has taught a computer system as described in claim 10. Furthermore, claim 18 is rejected for the same reasons set forth in the rejection of claim 9.

28. Referring to claim 19, Batten has taught a computer system as described in claim 10. Batten has further taught that the dependency descriptor includes live-in and live-out data for the dependency information. See Fig.9 and Fig.8, for instance. If the dependency descriptor

specifies that a WAR hazard does not exist, then the descriptor includes aggregate live-in data. That is, the descriptor comprises data which specifies that a hazard will not occur involving live-in data. And, in Fig.10, this is the case because s5 is a live-in value (from Fig.9, assuming the trace in Fig.10 follows that of Fig.9) and there is no hazard involving s5. In addition, if the dependency descriptor specifies that a RAW hazard exists, then the descriptor includes aggregate live-out data. That is, the descriptor comprises data which specifies that a hazard will occur involving live-out data. And, in Fig.9, this is the case because s5 is a live-out value (assuming the trace in Fig.10 follows that of Fig.9) and there is a hazard involving s5. Note that applicant has not defined within the claim what is meant by aggregate live-in and live-out data and consequently, because the examiner must take the broadest interpretation of the claim language, any data which involves live-in and live-out values is aggregate live-in and live-out data, respectively.

29. Referring to claim 20, Batten has taught a method of processing instructions comprising:

a) fetching a trace descriptor. See Fig.9 and note the ccdd instruction (trace descriptor). Being an instruction, it is inherently fetched.

b) separating out a dependency descriptor including dependency information for a set of instructions from the trace descriptor. Note the format of the trace descriptor in Fig.8. The dtype field represents a dependency descriptor which specifies the types of dependencies found within the instruction sequence. The descriptor is separated out when the system needs to determine what type of dependencies exist.

c) fetching the set of instructions described by dependency descriptor. Note from Fig.9, for instance, that each of the instructions in the instruction sequence will inherently be fetched before they are executed.

d) executing a plurality of the instruction sequences according to the dependencies stored in the dependency descriptor. See column 6, lines 44-58, and column 3, lines 52-57. Note that the dependency information is used by the hardware in executing the instructions such that stalls may be avoided by disabling the components which check for stalls.

30. Referring to claim 21, Batten has taught a method as described in claim 20. Batten has further taught updating live-out data in a first storage area. It is known that live-out data is data which is generated by a given trace (instruction sequence) that is needed by another trace for input. Looking at Fig.9-10, for instance, assume that the trace of Fig.10 follows the trace of Fig.9. It can be seen that r9, s5, and s3, of Fig.9 are live-out values because they are generated by the trace of Fig.9 and they are used as inputs by the trace of Fig.10. And, clearly these values are stored within registers, which are a fourth storage.

31. Referring to claim 22, Batten has taught a method as described in claim 21. Batten has further taught:

a) storing the dependency descriptor extracted by the control flow logic into a second storage area. See Fig.7, and note that the dependency descriptor, which is ultimately extracted, is stored in the memory storage associated with the dtype field.

b) reading the descriptor out of the second storage area into the data flow logic. Clearly, the information regarding the dependencies is read so that the hardware knows how to go about executing the associated instruction sequence.

32. Referring to claim 28, Batten has taught a machine-readable medium that provides instructions, which when executed by a machine cause the machine to perform operations comprising:

a) fetching a trace descriptor. See Fig.9 and note the ccdd instruction (trace descriptor). Being an instruction, it is inherently fetched.

b) separating out a dependency descriptor including dependency information for a set of instructions from the trace descriptor. Note the format of the trace descriptor in Fig.8. The dtype field represents a dependency descriptor which specifies the types of dependencies found within the instruction sequence. The descriptor is separated out when the system needs to determine what type of dependencies exist.

c) fetching the set of instructions described by dependency descriptor. Note from Fig.9, for instance, that each of the instructions in the instruction sequence will inherently be fetched before they are executed.

d) executing a plurality of the instruction sequences according to the dependencies stored in the dependency descriptor. See column 6, lines 44-58, and column 3, lines 52-57. Note that the dependency information is used by the hardware in executing the instructions such that stalls may be avoided by disabling the components which check for stalls.

33. Referring to claim 29, Batten has taught a method as described in claim 28. Batten has further taught that the operations further comprise updating live-out data in a first storage area. It is known that live-out data is data which is generated by a given trace (instruction sequence) that is needed by another trace for input. Looking at Fig.9-10, for instance, assume that the trace of Fig.10 follows the trace of Fig.9. It can be seen that r9, s5, and s3, of Fig.9 are live-out values

because they are generated by the trace of Fig.9 and they are used as inputs by the trace of Fig.10. And, clearly these values are stored within registers, which are a fourth storage.

34. Referring to claim 30, Batten has taught a method as described in claim 29. Batten has further taught:

a) storing the dependency descriptor extracted by the control flow logic into a second storage area. See Fig.7, and note that the dependency descriptor, which is ultimately extracted, is stored in the memory storage associated with the dtype field.

b) reading the descriptor out of the second storage area into the data flow logic. Clearly, the information regarding the dependencies is read so that the hardware knows how to go about executing the associated instruction sequence.

Claim Rejections - 35 USC § 103

35. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

36. Claims 4-5 and 13-14 are rejected under 35 U.S.C. 103(a) as being unpatentable over Batten, as applied above.

37. Referring to claim 4, Batten has taught a logic circuit as described in claim 3. Batten has further taught that a block of instructions may execute serially. See column 6, line 64, to column 7, line 4. Batten has not explicitly taught a third storage area coupled to the data flow logic, the third storage area to store instructions contiguously based on dependency information. However,

Official Notice is taken that the process of fetching using a program counter is well known and expected in the art. More specifically, a program counter (PC) holds an address at which a current instruction is fetched. After the current instruction is fetched, the PC is updated such that it points to the very next instruction which is located at an address immediately subsequent to the address associated with the current instruction. A person of ordinary skill in the art would have recognized that if a block of instructions is to execute serially (one after the other), as taught by Batten, then these instructions would have to be fetched in a serial manner. Using the PC, this is the typical way that instructions are fetched. Consequently, the instructions would have to be stored contiguously in a storage area since the PC points to contiguous memory locations. Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Batten to include store instructions contiguously based on dependency information so that they may be easily fetched for serial execution using a PC, which inherently exists in all systems.

38. Referring to claim 5, Batten has taught a logic circuit as described in claim 4. Batten has further taught a fourth storage area coupled to the data flow logic and control flow logic, the fourth storage area to store live-out data. It is known that live-out data is data which is generated by a given trace (instruction sequence) that is needed by another trace for input. Looking at Fig.9-10, for instance, assume that the trace of Fig.10 follows the trace of Fig.9. It can be seen that r9, s5, and s3, of Fig.9 are live-out values because they are generated by the trace of Fig.9 and they are used as inputs by the trace of Fig.10. And, clearly these values are stored within registers, which are a fourth storage.

39. Referring to claim 13, Batten has taught a computer system as described in claim 12. Furthermore, claim 13 is rejected for the same reasons set forth in the rejection of claim 4.

40. Referring to claim 14, Batten has taught a computer system as described in claim 13. Furthermore, claim 14 is rejected for the same reasons set forth in the rejection of claim 5.

41. Claims 6-7 and 15-16 are rejected under 35 U.S.C. 103(a) as being unpatentable over Batten, as applied above, in view of Sullery, "Beyond ILP II: SMT and Variants," May 2000 (herein referred to as Sullery).

42. Referring to claim 6, Batten has taught a logic circuit as described in claim 5. Batten has not taught a fifth storage area coupled to the control flow logic, the fifth storage area to map live-in and live-out data. However, Sullery has taught such a concept. See section 2.1 (paragraph beginning with "On trace dispatch..."). As is known in the art, such a map is used to implement register renaming, which is a process used to avoid WAW and WAR hazards. By avoiding hazards, instruction execution speed will be improved, thereby creating a more efficient system. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Batten to include a fifth storage which maps live-in and live-out values.

43. Referring to claim 7, Batten in view of Sullery has taught a logic circuit as described in claim 6. Batten has further taught that each of the storage areas are in at least one memory device. This is inherent because any component which includes a storage area is a memory device.

44. Referring to claim 15, Batten has taught a computer system as described in claim 14. Furthermore, claim 15 is rejected for the same reasons set forth in the rejection of claim 6.

45. Referring to claim 16, Batten has taught a computer system as described in claim 15. Furthermore, claim 16 is rejected for the same reasons set forth in the rejection of claim 7.

46. Claims 23-24 are rejected under 35 U.S.C. 103(a) as being unpatentable over Batten, as applied above, and further in view of Arimilli et al., U.S. Patent No. 6,427,204 (herein referred to as Arimilli).

47. Referring to claim 23, Batten has taught a method as described in claim 22. Batten has not taught that the fetching of a set of instructions is completed just in time for execution. However, Arimilli has taught such a concept. See column 3, lines 1-17. Note that Arimilli has taught that this is a more efficient way of fetching because instructions are only delivered when they are actually needed and pipeline bubbles are prevented. Consequently, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Batten such that instructions are fetched just-in-time, as taught by Arimilli.

48. Referring to claim 24, Batten in view of Arimilli has taught a method as described in claim 23. Although Batten has not taught that the instructions are out of order, Arimilli has taught such a concept. See column 1, line 61, to column 2, line 6. Note that the use of resources and efficiency are maximized with out-of-order execution. Consequently, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Batten to include instructions that are out-of-order, as taught by Arimilli.

49. Claims 25-26 are rejected under 35 U.S.C. 103(a) as being unpatentable over Batten in view of Arimilli, as applied above, in view of Witt et al., U.S. Patent No. 6,018,798 (herein referred to as Witt).

50. Referring to claim 25, Batten in view of Arimilli has taught a method as described in claim 24. Batten in view of Arimilli has not explicitly taught updating the architectural state using the data in the first storage area. However, Witt has taught the concept of having a speculative register file (future file 88, Fig.3) and an actual register file (Fig.3, component 102). The speculative register file holds the most current state of the machine (values determined via speculative execution) and by doing this, instructions may be executed speculatively. Once it is determined that instructions are no longer speculative, the speculative results are made architectural results by writing them to the actual register file. See column 12, line 66, to column 13, line 45. This is a known concept in the art. In essence, this scheme allows for speculative execution which is a method of executing instructions before it is known that they should execute (they are predicted to execute). This maximizes efficiency if they indeed were to execute (predicted correctly). As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Batten in view of Arimilli such that the architectural state is updated using the data in the speculative storage.

51. Referring to claim 26, Batten in view of Arimilli and further in view of Witt has taught a method as described in claim 25. Witt has further taught recovering an earlier architectural state after a misprediction using the data in the first storage area. See column 18, lines 54-67, and note that after a misprediction, a previous state is achieved by copying actual values into the

future file (so that the speculative values are correct). Consequently, by using this newly written data, the system recovers an earlier architectural state.

52. Claim 27 is rejected under 35 U.S.C. 103(a) as being unpatentable over Batten, as applied above, and further in view of Rotenberg et al., "A Trace Cache Microarchitecture and Evaluation," 1998 (as disclosed by applicant and herein referred to as Rotenberg).

53. Referring to claim 27, Batten has taught a method as described in claim 20. Batten has not taught that the selecting involves predicting the next trace descriptor to process. However, Rotenberg has taught such a concept. See page 3, and note the first paragraph under Figure 2. Rotenberg has taught that predicting sequences of traces for the implicit achievement of high branch prediction throughput. As a result, in order to increase branch prediction throughput, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Batten to include trace prediction as taught by Rotenberg.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to David J. Huisman whose telephone number is (703) 305-7811. The examiner can normally be reached on Monday-Friday (8:00-4:30).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (703) 305-9712. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

DJH
David J. Huisman
August 17, 2004

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100